Decoupled Interpolation-based Method for Numerical Simulation of Digital Controllers

Mehran Jafari*, Gautier Bureau†, Marco Chiaramello†, Adrien Guironnet†, Patrick Panciatici†, and Petros Aristidou*
*Dept. of Electrical Eng., Computer Eng., & Informatics, Cyprus University of Technology, Limassol, Cyprus
†Réseau de Transport d' Électricité (RTE), France
Corresponding email: mm.jafari@edu.cut.ac.cy

***Abstract*—All modern electric power systems employ digital controllers to ensure their secure and optimized operation. The time-domain simulation of such systems requires the solution of a set of hybrid (both continuous and discrete variables) differential-algebraic equations. This is not an easy task, since the discontinuities introduced by the digital controllers lead to constant time-step reductions and an increased computational burden. A method that is able to effectively perform such simulations with no time step reduction is the interpolation-based method (IBM). In this paper, we extend the IBM into the decoupled interpolation-based method to increase the computational performance while maintaining the same high simulation accuracy. The performance and accuracy of the proposed methods is employed on several small and medium system models.**

***Index Terms*—digital controllers, time-domain simulations, interpolation, decoupled simulation.**

I. Introduction

Digital controllers have become an important part of each modern dynamical system. Their purpose is to drive the systems to operate more reliably, economically, and sustainably. Time-domain simulations provide insights into the system security and reliability, which are very important indicators when assessing or planning the system operation.

Differential-algebraic equations (DAE) are frequently used to model physical systems, such as electrical power systems [1, 2]. Digital controllers in these systems are usually modeled as discrete models with difference equations to obtain the most accurate, close to reality, system response. However, this leads to the introduction of a discontinuity per each sampling action of digital controllers, therefore, making large-scale time-domain simulation of such systems a challenging task [3].

Overall, discontinuities or events can be categorized into *state events* and *time events* [4]. Events with their time of occurrence depending on state variables of the system are called *state events* while the events that happen in preknown points of time are *time events*. A specific component of the system reaching its critical temperature point can be an example of *state event* that its occurrence time is unknown in advance. However, digital controller sampling and actions are typical examples of *time event*, which their sampling times known a priori based to the controller sampling rate.

The most accurate method for treating any type of discontinuity in the time-domain simulation is the step reduction method (SRM). In this way, the time step is reduced to land exactly on the event [3], the changes enforced by the discontinuity are applied, and the simulation restarts. However, SRM forces a heavy computation burden on the solver due to the constant step size reduction, which leads to increased simulation time for systems with many digital controllers. In time-critical applications (such as dynamic security assessment), these delays can put the system at risk.

To tackle this problem, simplified simulation methods can be used to handle the discrete events. One such method, proposed in [5], assumes that the event occurs exactly at the end of the time step. Consequently, there is no need for time step size reduction. Although this method speeds up the simulation, the accuracy is sacrificed.

The interpolation-based method (IBM) [6], recently proposed by the authors, is able to simulate DAE systems alongside digital controllers modeled by difference equations without reducing the time step. As a result, the accuracy is maintained similar to SRM while its performance is close to the simplified simulation method. In this paper, we extend the IBM, proposing the decoupled interpolation-based method (DIBM) that solves the system DAEs separately from the digital controller equations to further accelerate the computation performance. Decoupling the controlled system equations from the controller equations and solving them separately results in a more frequent updating of the state variables and controller outputs, leading to faster convergence.

The rest of the paper is organized as follows. Section II briefly introduces the SRM and IBM methods for treating discrete events in a system. Then, the DIBM is proposed in Section III. Three case studies are utilized in Section IV to showcase the accuracy and performance of the proposed method, followed by the conclusions drawn in Section VI.

II. SRM and IBM Methodologies

In this section, the two different methods SRM and IBM used for handling the time events of digital controllers are briefly introduced.

First, we assume a physical dynamical system controlled by a digital controller, as shown in Fig. 1. The continuous system

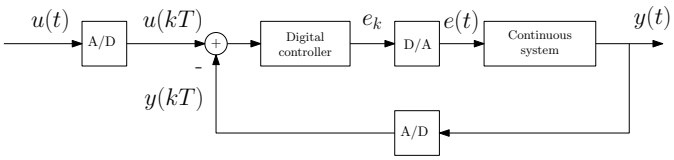

Fig. 1: A continuous system controlled by a digital controller

is modeled with Differential Algebraic Equations (DAEs) and can be formulated as follows:

$$\begin{aligned} \mathbf{0} &= \boldsymbol{F}(\dot{\boldsymbol{y}}(t), \boldsymbol{y}(t), e(t)) \\ \boldsymbol{y}(0) &= \boldsymbol{y}_0, \ e(0) = e_0 \end{aligned} \tag{1}$$

where $\boldsymbol{y}(t)$ is the vector of continuous state variables of the system and $e(t)$ the output of the controller to the system which can be calculated based on the input $u(t)$ and the feedback from the system $y(kT)$. The Analog-to-digital (A/D) converter samples the continuous states of the system every sample period T and the monitored states $\boldsymbol{y}(kT)$ are input to the digital controller.

The digital controller is usually modeled with difference equations formulated as follows:

$$e_k = \zeta(e_{k-1}, \boldsymbol{y}(kT)) \tag{2}$$

where e_k is the output of the digital controller at time $t \in [kT, (k+1)T]$, ζ denotes the controller function that depends on the historical output of the controller e_{k-1} and the quantized state variables monitored by the controller $\boldsymbol{y}(kT)$ at each sampling time kT. Finally, the Digital-to-Analog (D/A) converter transforms the control signal e_k to be used in the system as $e(t)$.

During the time-domain simulation of the DAE system (1), $\boldsymbol{y}(t_n)$ is the system solution at the n-th time step with its size h_n equal to $h_n = t_n - t_{n-1}$. The step size would usually be kept as large as possible, limited only by the numerical integration errors, to increase the simulation performance while maintaining high accuracy. However, time events are introduced by the digital controller sampling and action period, every T seconds, which are different from the time-step instances t_n. Furthermore, when multiple digital controllers with different sampling periods are acting in the same system, multiple discontinuities are inserted in the system simulation at irregular time intervals. Thus, proper methods are necessary to treat these discontinuities during the time-domain simulation.

A. Step reduction method (SRM)

Figure 2 shows the simulation of the system (1), where h_n is the time step that would be taken if the system were continuous without time events. However, due to the digital controller, the first time event within the time step h_n is at time $t_{n,1}$. Thus, the SRM adjusts the step to the reduced time step h_n^* to *land* exactly on the first discontinuity [4]. This process is repeated for each consecutive time event.

After the time step adjustment, the controller output $e(k)$ at $t_{n,1}$ is calculated using (2), then it is converted to a continuous signal $e(t_{n,1})$, and used with the continuous system (1) to find the solution $y(t_{n,1})$. It should be noted that the controller output is kept constant between the samples through a Zero-Order-Hold (ZOH) approach.

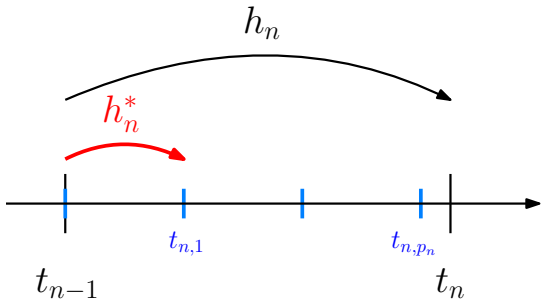

Fig. 2: Schematic of treating a discontinuity using SRM. The black vertical lines denote the simulation time steps, while the blue ones show the digital controller sampling time.

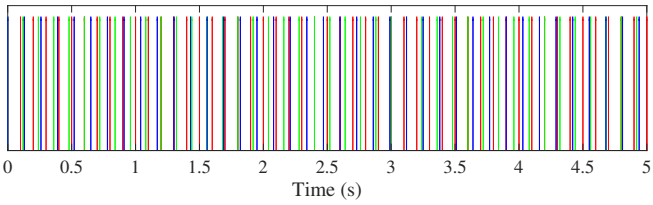

Fig. 3: Time event representation of three digital controllers with sampling times equal to 0.1 (red), 0.12 (green), 0.13 s (blue) for a 5 seconds simulation

SRM is the most accurate and widely used method, with its trajectory being the closest to the real operation of a digital controller. Each discontinuity is handled individually. However, since the step size is limited to the distance between every two time events, imposed by the digital controller sampling period, SRM constrains the time-step size and is computationally heavy. This can lead to very slow simulations in the case of a system with many digital controllers. For instance, the time events that stem from the operation of three digital controllers with various sampling times are illustrated in Fig. 3. Each bar represents one sample, and three colors of red, green, and blue are used to show the samplings of the different controllers. Other colors (if seen) indicate samples of two or more controllers overlapping. One can see that the time steps can be severely limited even for this very reduced number of digital controllers [6].

B. Interpolation-based method

Contrary to SRM, the IBM was proposed by the authors in [6] that computes the controller output e_k without the need to reduce the simulation time step. This is made possible by interpolating the system monitored variables $\boldsymbol{y}(t)$ at each controller sampling time within a time-step h_n to obtain $\boldsymbol{y}(kT)$ which is then used for the calculation of the controller output (see (2)). The controller output for all the sampling actions k within time-step h_n are incorporated into the state

variable vector of the system to be computed along the system state variables in each Newton iteration [6].

To describe it better, let's define the vector of the system's state variables at the n-th time-step by:

$$\boldsymbol{z}_n^1 = \boldsymbol{y}_n \tag{3}$$

and the vector of all the controllers' outputs within the time-step h_n by:

$$\boldsymbol{z}_n^2 = \begin{bmatrix} e_{n,1} & e_{n,2} & \dots & e_{n,g} & \dots & e_{n,p_n} \end{bmatrix}^T \tag{4}$$

where g denotes the index of the controller's output and p_n is the total number of controller samples within the time-step n. Combining the two, the new state variable vector can be formulated as follows:

$$\boldsymbol{z}_n = \begin{bmatrix} \boldsymbol{z}_n^1 \\ \boldsymbol{z}_n^2 \end{bmatrix} \tag{5}$$

In order to effectively solve for the new redefined state variable vector, we need to form also a new redefined residual function vector that consists of both residuals of the system's equations and controller's. This can be formulated as follows:

$$\tilde{\boldsymbol{g}} = \begin{bmatrix} \tilde{\boldsymbol{g}}_1 \\ \tilde{\boldsymbol{g}}_2 \end{bmatrix} \tag{6}$$

with:

$$\tilde{\boldsymbol{g}}_1(\boldsymbol{z}_n) = \boldsymbol{g}(\boldsymbol{y}_n, e_{n,p_n}) \tag{7}$$

$$\tilde{\boldsymbol{g}}_2(\boldsymbol{z}_n) = \begin{bmatrix} e_{n,1} - \zeta(e_{n,0}, \boldsymbol{x}_{n,1}) \\ \dots \\ e_{n,g} - \zeta(e_{n,g-1}, \boldsymbol{x}_{n,g}) \\ \dots \\ e_{n,p_n} - \zeta(e_{n,p_{n-1}}, \boldsymbol{x}_{n,p_n}) \end{bmatrix} \tag{8}$$

where $\boldsymbol{g}$ and $\tilde{\boldsymbol{g}}_1$ denote the residual function of the system equations, and $\tilde{\boldsymbol{g}}_2$ shows the residuals of controller outputs. $x_{n,g}^{(m)}$ refers to the interpolated values of the system state variables at $t_{n,g}$ as formulated below:

$$\boldsymbol{x}_{n,g}^{(m)} = \boldsymbol{w}_n^{(m)}(t_{n,g}), \quad \forall g \in [1, p_n] \tag{9}$$

where $w_n^{(m)}(t_{n,g})$ is the interpolation function and m denotes the index of Newton iteration.

Now, the new residual vector and the new state variable vector can be imported into the Newton solver to obtain the solution. The m-th Newton iteration to be solved is:

$$\boldsymbol{J}_n^{(m)}(\boldsymbol{z}_n^{(m+1)} - \boldsymbol{z}_n^{(m)}) = -\tilde{\boldsymbol{g}}(\boldsymbol{z}_n^{(m)}) \tag{10}$$

where $\boldsymbol{J}_n^{(m)}$ is the Jacobian matrix for both the system and controller equations, given by:

$$\boldsymbol{J}_n^{(m)} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,1}} & \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,2}} \\ \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,1}} & \frac{\partial \tilde{\boldsymbol{g}}_2}{\partial \boldsymbol{z}_{n,2}} \end{bmatrix} \tag{11}$$

The Jacobian matrix can be simplified without sacrificing accuracy by ignoring the impact of the system on the controller and vice versa (B and C), and assuming a unit vector for the impact of the controller on itself (D):

$$\boldsymbol{J}_n^{(m)} \approx \begin{bmatrix} \frac{\partial \tilde{\boldsymbol{g}}_1}{\partial \boldsymbol{z}_{n,1}} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_{p_n} \end{bmatrix} \tag{12}$$

Thus, leading to a dishonest Newton solution of (10).

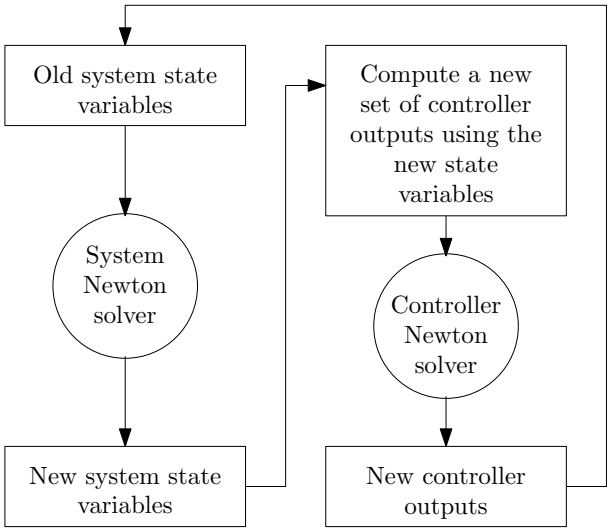

Fig. 4: A simple scheme showing DIBM

III. Decoupled interpolation-based method

In this section, the decoupled interpolation-based method (DIBM), as an extension of IBM, is described. The main idea is to solve the system and controller equations in two separate Newton loops and use the results of each one to update the inputs to the next iteration. This can be seen as an iterative decomposition method applied within the Newton iterations.

First, the system equations are solved, and the new solution is utilized to compute the new controller outputs. Then, the new controller outputs are used to start the next iteration and solve the new solution to the system. This process continues until both systems converge. A simplified scheme describing DIBM is illustrated in Fig. 4.

To better describe DIBM, let's again consider the state variables and controller outputs formulated as (3) and (4), respectively. In addition, the residuals are formulated as (7) and (8) for the system and controller equations, respectively. In the first stage, (3) and (7) result in the state variables and their associated mismatches used to solve the systems equations in the m-th Newton iteration as formulated in the following:

$$\boldsymbol{A}_n^{(m)}(\boldsymbol{z}_{n,1}^{(m+1)} - \boldsymbol{z}_{n,1}^{(m)}) = -\tilde{\boldsymbol{g}}_1(\boldsymbol{z}_{n,1}^{(m)}) \tag{13}$$

The state variables computed in the first stage trigger the second stage, where they are used for the interpolation function in (9) to obtain the interpolated state variables for each sampling time. Then, (4) and (8) result in a new set of controller outputs and associated mismatches, respectively.

Finally, the following Newton formula results in the corrected controller outputs for the next iteration:

$$\boldsymbol{D}_n^{(m)}(\boldsymbol{z}_{n,2}^{(m+1)} - \boldsymbol{z}_{n,2}^{(m)}) = -\tilde{\boldsymbol{g}}_2(\boldsymbol{z}_{n,2}^{(m)}) \tag{14}$$

This process repeats until the combined state variables vector z_n converges.

IV. Simulation Results

In this section, three test cases are used to showcase the advantages of the DIBM approach:

A. A general system consisting in a two-state continuous ODE system controlled by a digital integral controller.
B. A single-machine infinite-bus (SMIB) system containing a synchronous generator connected to an infinite bus under control by two digital controllers: a governor and an exciter.
C. The well-known Kundur test system which has 4 synchronous generators and eight digital controllers in total (two for each generator, in a similar way than test case B).

All test cases are also solved using SRM and IBM to provide an opportunity to compare the methods in terms of accuracy and performance. Finally, it is noteworthy that all the models and methods are developed in MATLAB [7] and all methods solve the same set of DAE equations for each test case.

A. Continuous system with an integral controller

For the first test system, a two-state continuous ODE system controlled by an integral controller is considered as follows:

$$\underbrace{\begin{bmatrix}\dot{y_1}\\ \dot{y_2}\end{bmatrix}}_{\dot{\boldsymbol{y}}} = \underbrace{\begin{bmatrix}a & b\\ -b & 0\end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix}y_1\\ y_2\end{bmatrix}}_{\boldsymbol{y}} + \underbrace{\begin{bmatrix}-b\\ 0\end{bmatrix}}_{\boldsymbol{B}} e(t) \tag{15}$$

and the integral controller with the gain $K_I = 0.07$ and sampling period $T = 0.1$ s is defined as follows:

$$e_k = e_{k-1} + K_I T(u(kT) - y_2(kT)) \tag{16}$$

The trajectories for the system's second output simulated for 50 s are depicted in Figs. 5 and 6 for a stable and unstable case (different a and b values) using all three methods (SRM, IBM, and DIBM). As can be seen, the accuracy of all three methods are the same for this case study.

Furthermore, the performance of all three methods is showcased in Table I in terms of the total number of Newton iterations for all time steps. It can be seen that the IBM-based methods are computationally faster than the SRM due to the time-step reduction. Comparing IBM with DIBM, the unstable case simulation shows no difference, while the simulation for the stable case is slightly faster.

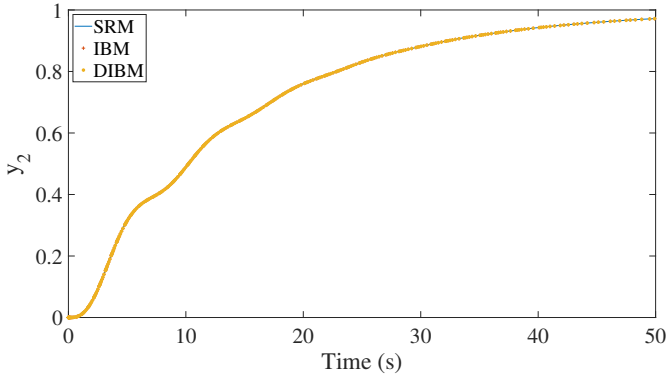

Fig. 5: Output results for the stable system controlled by an integral controller

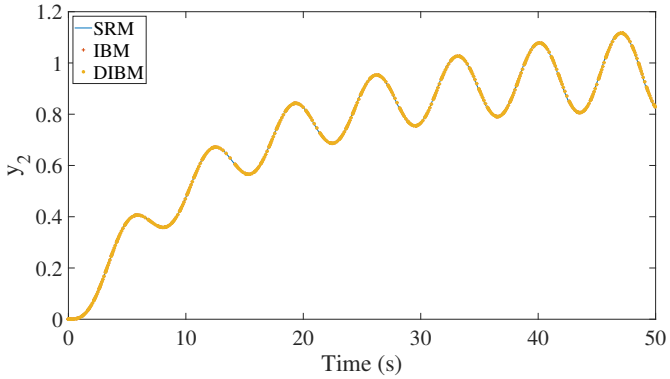

Fig. 6: Output results for the unstable system controlled by an integral controller

TABLE I: Performance result of the first case study simulations in terms of number of Newton iterations using SRM, IBM, and DIBM

System state	SRM	IBM	DIBM
Stable	4106	1071	1061
Unstable	3999	1758	1758

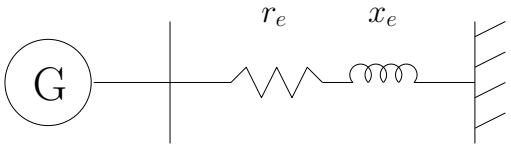

Fig. 7: Schematic of the single-machine infinite-bus system

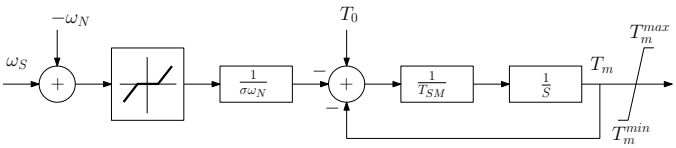

Fig. 8: Block diagram of the digital governor

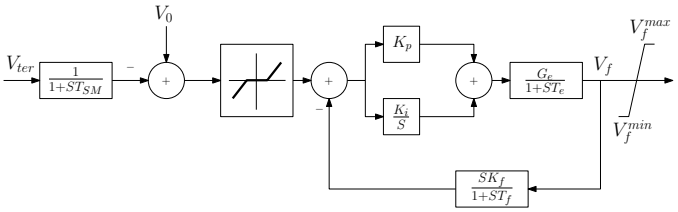

Fig. 9: Block diagram of the digital exciter

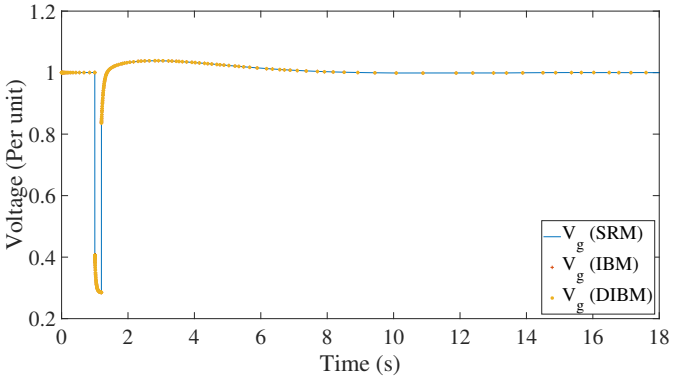

Fig. 10: Generator voltage output of SMIB system

TABLE II: Performance result of the second case study simulations in terms of number of Newton iterations and run time using SRM, IBM, and DIBM

Method	Nb. Newton iterations	Average of 5 run times (s)
SRM	4801	4.40
IBM	643	0.95
DIBM	624	0.88

B. SMIB test system

This test system consists in a synchronous generator connected to an infinite bus using a transmission line, as shown in Fig. 7. The generator is controlled by a digital governor [8] with a sampling period $T_G = 0.2$ s and a digital exciter [9] with a sampling period $T_G = 0.4$ s. The block diagrams are shown in Figs. 8 and 9, respectively.

A short circuit at $t = 1$ s is applied to the infinite bus for 200 ms. The generator voltage, the governor output and the exciter output are shown in Figs. 10, 11, and 12. Again, the same level of accuracy can be observed for all three methods.

The performance comparison is shown in Table II, listing the total number of Newton iterations and the execution time. The reported time is the average of 5 runs of the simulation to make sure of the validity of numbers. Again, SRM shows clearly the least performance while DIBM is slightly faster than IBM. The reason is that the controller outputs are calculated in each iteration with the updated state variables, which leads to faster convergence for some time steps.

C. Kundur test system

For the last case study, the well-known Kundur system [10], shown in Fig. 13, is considered. This electrical network

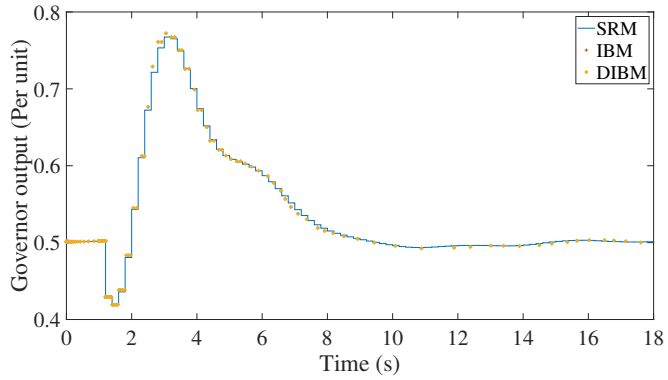

Fig. 11: Governor output of SMIB system

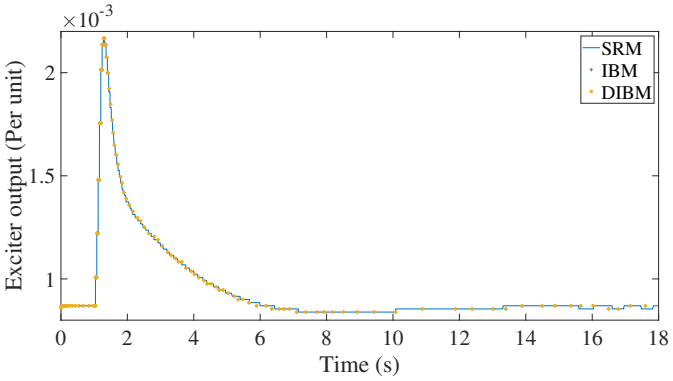

Fig. 12: Exciter output of SMIB system

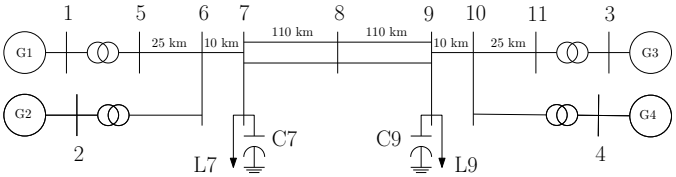

Fig. 13: Schematic of the Kundur test system

includes four synchronous generators, each under control by one digital governor and one digital exciter, and two loads that are connected with transmission lines of various lengths. The generator and controller models are identical to the previous case study B. The sampling periods are equal to 210, 220, 230, and 240 ms for the governors, and 41, 42, 43, and 44 ms for the exciters, respectively.

A load reduction of 250 MW in L7 is simulated at $t = 1$ s and then restored at $t = 12$ s. The simulation results for the voltage of buses 1, 4, and 9, the governor outputs, and the exciter outputs are illustrated in Figs. 14, 15, and 16.

Once more, the trajectories show the same accuracy between all three methods. However, the performance results listed in Table III demonstrate that the number of iterations for DIBM is slightly less than IBM and consequently the execution time for DIBM is smaller.

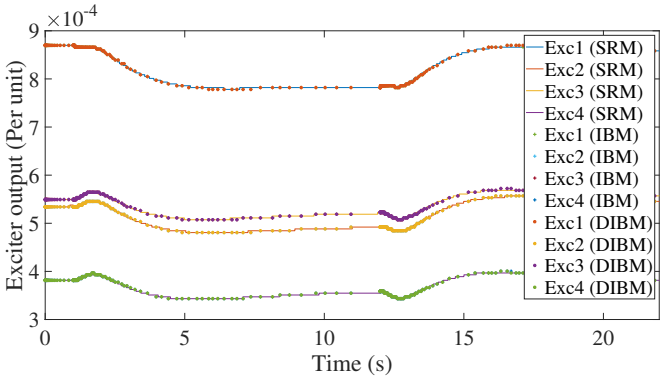

Fig. 16: Exciter outputs of Kundur system

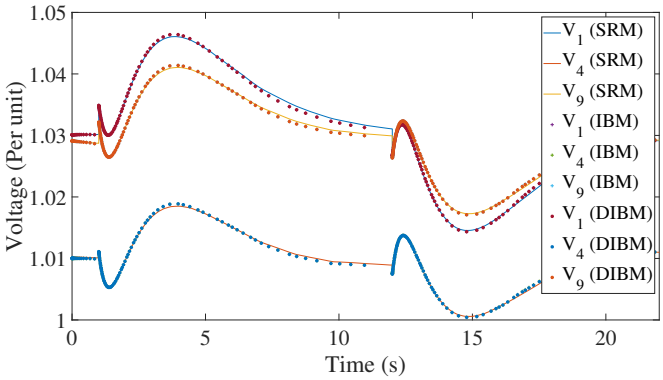

Fig. 14: Voltage of bus 1, 4, and 9 of Kundur system

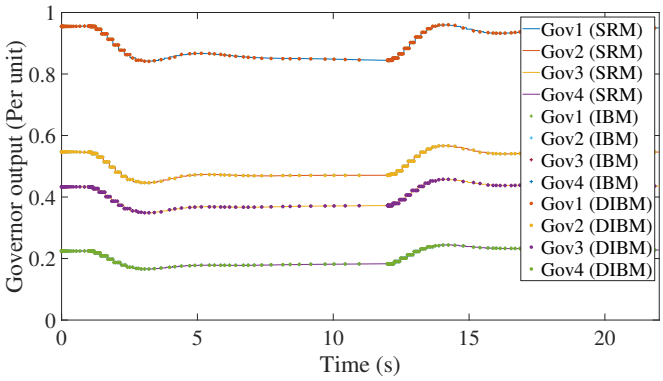

Fig. 15: Governor outputs of Kundur system

TABLE III: Performance result of the third case study simulations in terms of number of Newton iterations and run time using SRM, IBM, and DIBM

Method	Nb. Newton iterations	Average of 5 run times (s)
SRM	11927	42.48
IBM	604	2.54
DIBM	598	2.38

V. Discussion

In IBM, the controller actions are computed based on the previous Newton solution, while the controller outputs in DIBM are computed based on the intermediate (updated) solution which leads to a small decrease in the number of Newton iterations during the simulation. DIBM is also easier to implement since the size of sub-matrix A used for the first Newton solver is always constant and only the size of the identity matrix used for the second Newton solver varies in each time step while the Jacobian matrix for IBM containing both A and D needs to be formed unified and its size varies in each time step depending on the number of controller actions. However, the impact of the controller on the system and vice versa (sub-matrices B and C) cannot be considered in DIBM.

VI. Conclusion

In this paper, a novel method called the decoupled interpolation-based method which is based on the interpolation-based method was proposed. Similar to IBM, DIBM is capable of handling discrete events imposed on the simulation by the digital controller without reducing the step size. This method decouples the equations of the system from the digital controller and solves them separately.

Three test cases were used to showcase the accuracy and performance of DIBM against IBM, and SRM as the reference trajectory. The results show similar accuracy of DIBM compared to SRM and IBM while having the best performance in terms of the number of Newton iterations and simulation execution time.

For future work, the proposed methods will be incorporated into open-source simulation software to enable the use of large-scale test systems to better compare the methods. Also, the impact of other simplifications related to sub-matrices B and C on the method will be investigated.

References

[1] P. Aristidou, D. Fabozzi, and T. Van Cutsem, "Dynamic simulation of large-scale power systems using a parallel Schur-complement-based decomposition method," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 10, pp. 2561–2570, 2013.

[2] F. Milano, "Semi-implicit formulation of differential-algebraic equations for transient stability analysis," *IEEE Transactions on Power Systems*, vol. 31, no. 6, pp. 4534–4543, 2016.

[3] M. Jafari, G. Bureau, M. Chiaramello, A. Guironnet, P. Panciatici, and P. Aristidou, "Modeling of Digital Controllers in Electric Power System Dynamic Simulations," in *2023 IEEE Belgrade PowerTech*, 2023.

[4] D. Ellison, "Efficient automatic integration of ordinary differential equations with discontinuities," *Mathematics and Computers in Simulation*, vol. 23, no. 1, pp. 12–20, 1981.

[5] D. Fabozzi and T. Van Cutsem, "Simplified time-domain simulation of detailed long-term dynamic models," in *2009 IEEE Power & Energy Society General Meeting*. IEEE, 2009, pp. 1–8.

[6] M. Jafari, G. Bureau, M. Chiaramello, A. Guironnet, P. Panciatici, and P. Aristidou, "An Interpolation-based Method for Numerical Simulation of Digital Controllers in Power System Dynamic Studies," *TechRxiv. Preprint*, 2023. [Online]. Available: 10.36227/techrxiv.23579574

[7] T. M. Inc., *MATLAB version: 9.10.0 (R2021a)*. Natick, Massachusetts, United States: The MathWorks Inc., 2021. [Online]. Available: https://www.mathworks.com

[8] A. G. NEPLAN, "TURBINE-GOVERNOR MODELS: Standard Dynamic Turbine-Governor Systems in NEPLAN Power System Analysis Tool," Tech. Rep., 2015.

[9] "IEEE Recommended Practice for Excitation System Models for Power System Stability Studies," pp. 1–207, 2016.

[10] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-hill New York, 1994, vol. 7.